

NAG Toolbox for MATLAB

f12fe

1 Purpose

f12fe can be used to return additional monitoring information during computation. It is in a suite of functions consisting of f12fa, f12fb, f12fc, f12fd and f12fe.

2 Syntax

```
[niter, nconv, ritz, rzest] = f12fe(icommm, comm)
```

3 Description

The suite of functions is designed to calculate some of the eigenvalues, λ , (and optionally the corresponding eigenvectors, x) of a standard eigenvalue problem $Ax = \lambda x$, or of a generalized eigenvalue problem $Ax = \lambda Bx$ of order n , where n is large and the coefficient matrices A and B are sparse, real and symmetric. The suite can also be used to find selected eigenvalues/eigenvectors of smaller scale dense, real and symmetric problems.

On an intermediate exit from f12fb with **irevcm** = 4, f12fe may be called to return monitoring information on the progress of the Arnoldi iterative process. The information returned by f12fe is:

- the number of the current Arnoldi iteration;
- the number of converged eigenvalues at this point;
- the real and imaginary parts of the converged eigenvalues;
- the error bounds on the converged eigenvalues.

f12fe does not have an equivalent function from the ARPACK package which prints various levels of detail of monitoring information through an output channel controlled via a parameter value (see Lehoucq *et al.* 1998 for details of ARPACK routines). f12fe should not be called at any time other than immediately following an **irevcm** = 4 return from f12fb.

4 References

Lehoucq R B 2001 Implicitly Restarted Arnoldi Methods and Subspace Iteration *SIAM Journal on Matrix Analysis and Applications* **23** 551–562

Lehoucq R B and Scott J A 1996 An evaluation of software for computing eigenvalues of sparse nonsymmetric matrices *Preprint MCS-P547-1195* Argonne National Laboratory

Lehoucq R B and Sorensen D C 1996 Deflation Techniques for an Implicitly Restarted Arnoldi Iteration *SIAM Journal on Matrix Analysis and Applications* **17** 789–821

Lehoucq R B, Sorensen D C and Yang C 1998 *ARPACK Users' Guide: Solution of Large-Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods* SIAM, Philadelphia

5 Parameters

5.1 Compulsory Input Parameters

1: **icommm**(*) – int32 array

Note: the dimension of the array **icommm** must be at least $\max(1, \mathbf{licomm})$, where **licomm** is passed to the setup function f12fa (see f12fa).

The array **icommm** output by the preceding call to f12fb.

2: **comm**(*) – double array

Note: the dimension of the array **comm** must be at least $\max(1, 3 \times \mathbf{n} + \mathbf{ncv} \times \mathbf{ncv} + 8 \times \mathbf{ncv} + 60)$ (see f12fa).

The array **comm** output by the preceding call to f12fb.

5.2 Optional Input Parameters

None.

5.3 Input Parameters Omitted from the MATLAB Interface

None.

5.4 Output Parameters

1: **niter** – int32 scalar

The number of the current Arnoldi iteration.

2: **nconv** – int32 scalar

The number of converged eigenvalues so far.

3: **ritz**(*) – double array

Note: the dimension of the array **ritz** must be at least **ncv** (see f12fa).

The first **nconv** locations of the array **ritz** contain the real converged approximate eigenvalues.

4: **rzest**(*) – double array

Note: the dimension of the array **rzest** must be at least **ncv** (see f12fa).

The first **nconv** locations of the array **rzest** contain the Ritz estimates (error bounds) on the real **nconv** converged approximate eigenvalues.

6 Error Indicators and Warnings

None.

7 Accuracy

A Ritz value, λ , is deemed to have converged if its Ritz estimate $\leq \mathbf{Tolerance} \times |\lambda|$. The default **Tolerance** used is the *machine precision* given by x02aj.

8 Further Comments

None.

9 Example

```
n = int32(100);
nx = int32(10);
nev = int32(4);
ncv = int32(10);

irevcm = int32(0);
resid = zeros(100,1);
v = zeros(100,20);
```

```

x = zeros(100,1);
mx = zeros(100,1);

sigma = 0;

% Initialisation Step
[icomm, comm, ifail] = f12fa(n, nev, ncv);

% Set Optional Parameters
[icomm, comm, ifail] = f12fd('SMALLEST MAGNITUDE', icomm, comm);

% Solve
while (irevcn ~= 5)
    [irevcn, resid, v, x, mx, nshift, comm, icomm, ifail] = ...
        f12fb(irevcn, resid, v, x, mx, comm, icomm);
    if (irevcn == 1 || irevcn == -1)
        x = f12f_av(nx, x);
    elseif (irevcn == 4)
        [niter, nconv, ritz, rzest] = f12fe(icomm, comm);
        fprintf('Iteration %d, No. converged = %d, norm of estimates = %16.8g\n', niter, nconv, norm(rzest(1:nev),2));
    end
end

% Post-process to compute eigenvalues/vectors
[nconv, d, z, v, comm, icomm, ifail] = f12fc(sigma, resid, v, comm, icomm)

```

```

Iteration 1, No. converged = 0, norm of estimates =      81.010211
Iteration 2, No. converged = 0, norm of estimates =      45.634095
Iteration 3, No. converged = 0, norm of estimates =      42.747772
Iteration 4, No. converged = 0, norm of estimates =       8.6106757
Iteration 5, No. converged = 0, norm of estimates =       0.71330195
Iteration 6, No. converged = 0, norm of estimates =       0.15050738
Iteration 7, No. converged = 0, norm of estimates =       0.015776765
Iteration 8, No. converged = 0, norm of estimates =      0.0038996544
Iteration 9, No. converged = 0, norm of estimates =      0.0004324447
Iteration 10, No. converged = 0, norm of estimates =      0.00011026365
Iteration 11, No. converged = 0, norm of estimates =      1.2358564e-05
Iteration 12, No. converged = 0, norm of estimates =      3.1712516e-06
Iteration 13, No. converged = 1, norm of estimates =      3.5633917e-07
Iteration 14, No. converged = 1, norm of estimates =      4.1370551e-08
Iteration 15, No. converged = 2, norm of estimates =      5.3820859e-09
Iteration 16, No. converged = 1, norm of estimates =       7.557502e-10
Iteration 17, No. converged = 1, norm of estimates =      4.0268792e-11
Iteration 18, No. converged = 2, norm of estimates =      1.4048495e-11
Iteration 19, No. converged = 2, norm of estimates =       1.774664e-10
Iteration 20, No. converged = 2, norm of estimates =      1.1754284e-09
Iteration 21, No. converged = 2, norm of estimates =       7.782271e-09
Iteration 22, No. converged = 2, norm of estimates =      5.2551985e-10
Iteration 23, No. converged = 2, norm of estimates =      1.8326785e-10
Iteration 24, No. converged = 2, norm of estimates =      7.8753868e-11
Iteration 25, No. converged = 2, norm of estimates =      2.2420618e-11
Iteration 26, No. converged = 2, norm of estimates =      8.5727233e-13
Iteration 27, No. converged = 2, norm of estimates =      3.4872081e-13
Iteration 28, No. converged = 2, norm of estimates =      2.6274361e-14
Iteration 29, No. converged = 2, norm of estimates =      2.1845559e-14
Iteration 30, No. converged = 3, norm of estimates =       8.670653e-16
Iteration 31, No. converged = 3, norm of estimates =      2.5548816e-16
Iteration 32, No. converged = 3, norm of estimates =      4.9338532e-18
nconv =

```

4

```

d =
    19.6054
    48.2193
    48.2193
    76.8333
     0
     0

```

```
          0
          0
          0
          0
z =
    array elided
v =
    array elided
comm =
    array elided
icomm =
    array elided
ifail =
          0
```
